QUIMERA - Toward an Unifying Quality Metamodel
Alfonso Garcia Frey, Eric Céret, Sophie Dupuy-Chessa, Gaëlle Calvary

To cite this version:
Alfonso Garcia Frey, Eric Céret, Sophie Dupuy-Chessa, Gaëlle Calvary. QUIMERA - Toward an Unifying Quality Metamodel. INFORSID 2011, May 2011, Lille, France. <hal-00761092>
QUIMERA

Toward an unifying quality metamodel

Alfonso García Frey — Eric Céret — Sophie Dupuy-Chessa — Gaëlle Calvary

University of Grenoble, CNRS, LIG UMR 5217 Grenoble
385 Avenue de la Bibliothèque, 38400, Saint Martin d'Hères, France

RÉSUMÉ. Différents modèles et méta-modèles de qualité existent pour les Systèmes d'Information, le Génie Logiciel ou l'Interaction Homme-Machine. Certains sont orientés vers l'évaluation de code et ne considèrent pas les modèles du système ; d'autres ne traitent pas des résultats de l'évaluation de la qualité. Enfin il leur manque parfois une des perspectives sur la qualité. Pour couvrir tous ces aspects, nous proposons QUIMERA, un méta-modèle de qualité qui unifie les exigences de qualité des différents domaines. Ce papier illustre QUIMERA sur deux cas d'études relevant de l'IHM et des systèmes d'information.

ABSTRACT. Different quality metamodels and quality models exist in both Information Systems, Software Engineering and Human-Computer Interaction. Some of these models are oriented to evaluating source code or final products and not models or modeling activities. Others simply don't deal with evaluation aspects (evaluation methods, results...) or they just miss the different quality perspectives. To cover these aspects, we propose QUIMERA, a quality metamodel for unifying both Software Engineering and Human Computer Interaction quality requirements. This paper illustrates QUIMERA on two case studies, representing quality models for Human-Computer Interaction and for an Information Systems.

MOTS-CLÉS : Meta-modèle de qualité, modèle de qualité, perspectives de la qualité, évaluation de la qualité, utilisabilité.

KEYWORDS: Quality Metamodel, quality model, Quality perspectives, quality evaluation, usability.
1. Introduction

Quality is an important concept that can make differences between products. We need to understand it in order to evaluate it in an automatic or not manner. Moreover nowadays, an automatic evaluation can be used from quality driven adaptation in context-aware applications to the generation of explanations about quality design choices. To support these new roles, we propose QUIMERA, a QAuality metamodel to IMprove the dEsign RAtonale. QUIMERA unifies quality aspects from Human-Computer Interaction (HCI), Informations Systems (IS) and Software Engineering (SE).

After having presented related quality metamodels in section 2, QUIMERA is described in section 3. We show its large spectrum with two examples (section 4). The first one comes from HCI. It shows a quality model for Ergonomic Criteria. The second one comes from the evaluation of an information system design method and shows a quality model for source code evaluation. Finally we conclude with some perspectives.

2. Related Works

Different quality metamodels and quality models exist in different domains. In SE, McCall’s hierarchical quality model (McCall et al., 1977) focuses on product quality, organizing it in two views: the external view for end-users and the internal view for developers. Boehm's model (Boehm et al., 1978) adds a third level named primitive characteristics to deal with metrics and evaluation. The ISO/IEC 9126 (ISO, 2001) standard series divides metrics into internal, external and quality-in-use that is related to HCI.

In IS, other quality metamodels have been proposed such as (Kashif et al., 2009) for data quality, (Mohagheghi et al., 2008) as a quality metamodel for Model-Driven Engineering (MDE), or (Dromey et al., 1996) that defines a five step process for building product-specific quality models.

However, whilst several quality models exist, most of them are oriented to evaluating source code or final products and not models or modeling activities. Other models don't deal with the results of quality evaluation or they just miss the

![Figure 1. Quality perspectives in QUIMERA](image-url)
different quality perspectives (expected, wished, ...). QUIMERA takes benefit from both the previously cited works while covering their requirements as well as our needs for user interface quality.

3. QUIMERA: The quality metamodel

3.1. Quality perspectives

QUIMERA (figure 1) has been designed to cover the four different perspectives in quality (Carlier, 2006):

– Expected Quality is the quality the client needs. It is defined through the specification of the System Under Study (SUS).

– Wished Quality is the degree of quality that the quality expert wants to achieve for the final version of the SUS. It is derived from the Expected Quality.

– Achieved Quality is the quality obtained for a given implementation of the SUS. Ideally, it must satisfy the Wished Quality.

– Perceived Quality is the perception of the quality by the client, once the SUS has been delivered.

As stated in (Si-Saïd Cherfi et al., 2002), these four perspectives can be related to the Systems Development Life Cycle by three dimensions. These dimensions are the Specification (related to the Expected and Wished Qualities), Implementation (related to the Achieved Quality) and Use (related to the Perceived Quality).

QUIMERA deals with these four perspectives as shown in figure 1. Here, the System entity represents the product to consider. SysEval represents a specific evaluation for that product. The four quality perspectives are four different uses of the same quality model. The attribute standard means that, when true, the quality model is not linked to System and SysEval as it only represents a quality standard such as ISO9126 or QUIM (Seffah et al., 2006). In other words, the quality of these standards is not defined in terms of a product. Some internal parts of QUIMERA are not necessarily defined when standard is true. Once the standard has been set, QUIMERA can be extended with the classes that are needed for each quality perspective, as we will see in the next section.

3.2. The metamodel

Figure 2 shows QUIMERA in detail. A quality model is composed of criteria, that can be recursively decomposed into subcriteria through the class
CriterionAssociation. Different recommendations can be specified for each criterion. A Recommendation is a positive assessment that characterizes Criteria. Evaluations can be performed through EvaluationMethods that are specified by Metrics and/or Practices. In case of Metrics, the measure is given by a NumericalResult that can be comprised between some Limits when those limits are defined. In the case of Practices, the result is a logical value, true or false, indicating if the Practice has been followed or not. Note that a Practice can be either a pattern or an anti-pattern, applied at the process level, or on a product. Metrics and Practices are directly evaluated on Artifacts through Recommendations. An Artifact can be any element of the Software Development Life Cycle, such as code, classes of a model or the model itself. Once a quality standard has been defined through Criteria, the metamodel can be reused with the association relatedTo, and extended.
with several classes such as EvaluationMethods, Transformations or Artifacts, to represent the four quality perspectives. For instance, Metrics can be defined in order to obtain some desired values (Wished Quality). The importance of every Recommendation can be customized using Weights. This allows designers to adjust the global quality precisely. Then, evaluations of the current quality of the SUS can be performed. When a Result of the evaluation (Numerical from Metrics or Logical from Practices) does not satisfy the expectations of the quality expert, that is, the Achieved Quality does not satisfy the Wished Quality, (for instance, the value for a metric is not within the desired Limits), the designer will need to increase the quality. This can be done by setting a Transformation or a set of Transformations. These Transformations are performed on the related Artifacts on which the Result has been previously calculated. Iterations can be done until the desired values defined by the quality expert (Wished Quality) are reached.

4. Case Studies

4.1. A quality model covering the ergonomic criteria in HCI

Figure 3 shows an excerpt of a quality model of Ergonomic Criteria in HCI (Bastien et al., 1993). The criteria are divided into subcriteria until the final ergonomic rules are derived. For the sake of brevity, we focus on three criteria:

– Error Protection is a subcriterion of Error Management. It refers to the means available to detect and prevent data entry errors or actions with destructive consequences.

– Minimal Actions is a subcriterion of Workload. It concerns workload with respect to the number of actions necessary to accomplish a task.

– Prompting is a subcriterion of Guidance. It refers to the means available in order to lead the users to make specifications, providing the required formats and values.

A Recommendation is a positive assessment that characterizes one or more criteria. Figure 3 shows how different metrics are used for the same recommendation. The recommendation says that good quality can be achieved by maximizing the number of criteria that are satisfied by a User Interface. To evaluate Criteria, two different EvaluationMethods are defined based on different formulas.

4.2. Application to the evaluation of code quality

Originally developed by the UMANIS Company, Symphony is a method focused on business components. It has been extended to include the design of complex interfaces (Godet-Bar et al., 2007). Symphony is based on the iterative
identification and description of business components. The extension of Symphony supports design of HCI concerns in a similar way: interactional entity objects are basic interactional concepts, i.e. the graphical representation of a concept. Interactional process objects describe the logic of the interactional domain, e.g. the management of an immersive 3D scene.

The purpose of the research described in (Céret et al, 2010) is to verify that the use of interactional and business objects, and the management of communication between all those components improve the final quality of the software. Thus, the quality of several implementations of the same project has been measured and compared, and software quality criteria and metrics have been defined and valued.

We have modeled these criteria and metrics according to QUIMERA. The resulting model contains 39 classes. Figure 4 presents a subset of these elements.

Two criteria, reusability and maintainability are visible here. These criteria are refined when needed, e.g. maintainability is composed of independence, sizes and complexity criteria. Recommendations are associated to criteria: according to (McCabe, 1996), we defined that the cyclomatic complexity - the number of linearly independent paths in the code, i.e. the minimum number of paths that should be
tested - has to be low so that the code can actually be tested. The different limits for each metric have been modeled: cyclomatic complexity is good when lower than 4, and too high when greater than 11 (Céret et al, 2010). The numerical results have been represented and associated to an artifact, here the whole application. In this application, we can say that QUIMERA was well adapted to describe the quality

![Diagram](image)

**Figure 4. Subset of the objects of Symphony evaluation model**

5. Conclusion

We propose QUIMERA, a quality metamodel that unifies quality aspects from HCI, IS and SE. To illustrate its genericity, we have applied QUIMERA to two case studies from different domains.

Future work will focus on implementing QUIMERA in a toolkit for using it at: 1) design time: making the quality explicit should help designers to take the right design choices; 2) at runtime: generating explanations about the Achieved Quality of a given User Interface should help end-users to understand why a User Interface is the way it is. Embedding quality models at runtime will allow us to explore new concepts such as quality driven adaptation of context aware applications, blurring the lines between designers, systems and end-users.
5. Bibliography


